Core Java Sample Assignments

1. Write a Java program to check whether a Number is Armstrong or not.

An Armstrong number is the one that equals the sum of its digits raised to the power of the number of digits in that number which is to be checked. To be more clear, let the number be n and the number of digits be x. We represent the number as nxnx-1nx-2...n3n2n1 where n1, n2, n3...nx are single digits 0-9. n is an Armstrong number if n1x + n2x + n3x + nx-1x + nxx = n
153, 371, 9474 and 54748 are few Armstrong numbers.
153 = 13 + 53 + 33
371 = 33 +73 +13
9474 = 94 + 44 +74 + 44
54748 = 55 + 45 + 75 + 45 + 85

```
import java.util.Scanner;
public class Test {
public static void main(String[] args) {
int num;
Scanner sc=new Scanner(System.in);
num=sc.nextInt();
int n = num; //use to check at last time
int check=0,remainder;
while(num > 0){
remainder = num % 10;
check = check +(remainder*remainder*remainder) ;
num = num / 10;
}
if(check == n)
System.out.println(n+" is an Armstrong Number");
else
System.out.println(n+" is not a Armstrong Number");
}
}
```

2. Write a java program to create an abstract class named Shape that contains two integers and an empty method named print Area(). Provide three classes named Rectangle, Triangle and Circle such that each one of the classes extends the class Shape. Each one of the classes contain only the method print Area( ) that prints the area of the given shape.

```
abstract class Shape
{
abstract void numberOfSides();
}
class Trapezoid extends Shape
{
void numberOfSides()
{
System.out.println(" Trapezoidal has four sides");
}
}
class Triangle extends Shape
{
void numberOfSides()
{
System.out.println("Triangle has three sides");
}
}
class Hexagon extends Shape
{
void numberOfSides()

{
System.out.println("Hexagon has six sides");
}
}
class ShapeDemo
{
public static void main(String args[ ])
{
Trapezoid t=new Trapezoid();
Triangle r=new Triangle();
Hexagon h=new Hexagon();
Shape s;
s=t;
s.numberOfSides();
s=r;
s.numberOfSides();
s=h;
s.numberOfSides();
```

```
}
}
```

3. Write a java program that implements Quick Sort Algorithm for sorting list names in ascending order.

```
/**
** Java Program to Implement Quick Sort
**/
import java.util.Scanner;
/** Class QuickSort **/
public class QuickSort
{
/** Quick Sort function **/
public static void sort(int[] arr)
{
quickSort(arr, 0, arr.length - 1);
}
/** Quick sort function **/
public static void quickSort(int arr[], int low, int high)
{
int i = low, j = high;
int temp;
int pivot = arr[(low + high) / 2];
/** partition **/
while (i <= j)
{
while (arr[i] < pivot)
i++;
while (arr[j] > pivot)
j--;
if (i <= j)
{
/** swap **/
temp = arr[i];
arr[i] = arr[j];
arr[j] = temp;
i++;
j--;
```

```
}
}
/** recursively sort lower half **/
if (low < j)
quickSort(arr, low, j);
/** recursively sort upper half **/
if (i < high)
quickSort(arr, i, high);
}
/** Main method **/
public static void main(String[] args)
{
Scanner scan = new Scanner( System.in );
System.out.println("Quick Sort Test\n");
int n, i;
/** Accept number of elements **/
System.out.println("Enter number of integer elements");
n = scan.nextInt();
/** Create array of n elements **/
int arr[] = new int[ n ];
/** Accept elements **/
System.out.println("\nEnter "+ n +" integer elements");
for (i = 0; i < n; i++)
arr[i] = scan.nextInt();
/** Call method sort **/
sort(arr);
/** Print sorted Array **/
System.out.println("\nElements after sorting ");
for (i = 0; i < n; i++)
System.out.print(arr[i]+" ");
System.out.println();
}
}
```

4. Write a Java program that implements bubble sort algorithm for sorting in descending order and also shows the number of interchanges occurred for the given set of integers.

```
public class BubbleSortDescendingOrderDemo
{
public static void main(String a[])
```

```
{
//Numbers which need to be sorted
int numbers[] = {23,5,23,1,7,12,3,34,0};
//Displaying the numbers before sorting
System.out.print("Before sorting, numbers are ");
for(int i = 0; i < numbers.length; i++)
{
System.out.print(numbers[i]+" ");
}
System.out.println();
//Sorting in descending order using bubble sort
bubbleSortInDescendingOrder(numbers);
//Displaying the numbers after sorting
System.out.print("Before sorting, numbers are ");
for(int i = 0; i < numbers.length; i++)
{
System.out.print(numbers[i]+" ");
}

}
//This method sorts the input array in desecnding order
public static void bubbleSortInDescendingOrder(int numbers[])
{
int temp;
for(int i = 0; i < numbers.length; i++)
{
for(int j = 1; j < (numbers.length-i); j++)
{
//if numbers[j-1] < numbers[j], swap the elements
if(numbers[j-1] < numbers[j])
{
temp = numbers[j-1];
numbers[j-1]=numbers[j];
numbers[j]=temp;
} } } }
```

5. Program to find the average of numbers using array

```
public class JavaExample {
public static void main(String[] args) {
double[] arr = {19, 12.89, 16.5, 200, 13.7};
double total = 0;
```

```
for(int i=0; i<arr.length; i++){
total = total + arr[i];
}
/* arr.length returns the number of elements
* present in the array
*/
double average = total / arr.length;
/* This is used for displaying the formatted output
* if you give %.4f then the output would have 4 digits
* after decimal point.
*/
System.out.format("The average is: %.3f", average);
}
}
```

6. Java program to demonstrate example of static variable and static method

```
import java.util.*;
class Item
{
private String itemName;
private int quantity;
private static int cnt=0; //variable to count
public void getItem()
{
Scanner sc=new Scanner(System.in);
System.out.print("Enter item name: ");
itemName=sc.next();
System.out.print("Enter item quantity: ");
quantity=sc.nextInt();
//increment counter
cnt++;
}
public void showItem()
{
System.out.println("Item Name: " + itemName + "\tQuantity: " +quantity);
}
public static int getCounter()
{
return cnt;
}
}
```

```
public class StaticVar
{
public static void main(String []s)
{
try
{
Item I1=new Item();
Item I2=new Item();
Item I3=new Item();
I1.getItem();
I2.getItem();
I3.getItem();
I1.showItem();
I2.showItem();
I3.showItem();
System.out.println("Total object created (total items are): "+ Item.getCounter());
}
catch (Exception e)
{
System.out.println(e.toString());
}
}
}
```

7. Java program to find count of all digits of a number using class

```
import java.util.*;
class DigitsOpr
{
private int num;
//function to get value of num
public void getNum(int x)
{
num=x;
}//End of getNum()
//function to count total digits
public int countDigits()
{
int n,count;
n=num; //keep value of num safe
count=0;//reset counter
while(n>0)
```

```
{
n/=10;
count++;
}
return count;
}//End of countDigits()
}
public class number
{
public static void main(String []s)
{
DigitsOpr dig=new DigitsOpr();
int n;
Scanner sc=new Scanner(System.in);
//read number
System.out.print("Enter an +ve integer number:");
n=sc.nextInt();
dig.getNum(n);
System.out.println("Total number of digits are: " + dig.countDigits());
}
}
```

8. Java program to get sub string from a given string

```
import java.util.*;
class getSubstring
{
public static void main(String args[])
{
Scanner sc=new Scanner(System.in);
String str="";
int startIndex,endIndex;
//input string
System.out.print("Enter the string: ");
str=sc.next();
//input start index and end index
System.out.print("Enter start index: ");
startIndex=sc.nextInt();
System.out.print("Enter end index: ");
endIndex=sc.nextInt();
/*get string from startIndex to endIndex*/
String temp;
```

```
temp= str.substring(startIndex, endIndex);
//printing substring
System.out.println("Substring is: "+temp);
}
}
```

9. Java Program to display first n or first 100 prime numbers

```
import java.util.Scanner;
class PrimeNumberDemo
{
public static void main(String args[])
{
int n;
int status = 1;
int num = 3;
//For capturing the value of n
Scanner scanner = new Scanner(System.in);
System.out.println("Enter the value of n:");
//The entered value is stored in the var n
n = scanner.nextInt();
if (n >= 1)
{
System.out.println("First "+n+" prime numbers are:");
//2 is a known prime number
System.out.println(2);
}
for ( int i = 2 ; i <=n ; )
{
for ( int j = 2 ; j <= Math.sqrt(num) ; j++ )
{
if ( num%j == 0 )
{
status = 0;
break;j
}
}
if ( status != 0 )
{
System.out.println(num);
i++;
}
```

```
status = 1;
num++; } }}
```

10. Fibonacci Series in Java without using recursion

```
class FibonacciExample1{
public static void main(String args[])
{
int n1=0,n2=1,n3,i,count=10;
System.out.print(n1+" "+n2);//printing 0 and 1
for(i=2;i<count;++i)//loop starts from 2 because 0 and 1 are already printed
{
n3=n1+n2;
System.out.print(" "+n3);
n1=n2;
n2=n3;
}
}
}
```

11. Write a Java Program that displays the number of characters, lines and words in a text file.

```
import java.io.*;
public class FileStat {
public static void main(String args[ ])throws IOException {
long nl=0,nw=0,nc=0;
String line;
BufferedReader br=new BufferedReader(new FileReader(args[0]));
while ((line=br.readLine())!=null) {
nl++;
nc=nc+line.length();
int i=0;
boolean pspace=true;
while (i<line.length()) {
char c=line.charAt(i++);
boolean cspace=Character.isWhitespace(c);
if (pspace&&!cspace)
nw++;
pspace=cspace;
}
}
System.out.println("Number of Characters"+nc);
```

```
System.out.println("Number of Characters"+nw);
System.out.println("Number of Characters"+nl);
}}
// Alternate solution using StringTokenizer
import java.io.*;

import java.util.*;
public class FileStat {
public static void main(String args[ ])throws IOException {
long nl=0,nw=0,nc=0;
String line;
BufferedReader br=new BufferedReader(new FileReader(args[0]));
while ((line=br.readLine())!=null) {
nl++;
nc=nc+line.length();
StringTokenizer st = new StringTokenizer(line);
nw += st.countTokens();
}
System.out.println("Number of Characters"+nc);
System.out.println("Number of Characters"+nw);
System.out.println("Number of Characters"+nl);
}}
```

12. Write a Java Program that reads on file name from the user, then displays information about whether the file exists, whether the file is readable, whetеher the file is writable, the type of file and the length of the file in bytes.

```
import java.io.File;
class FileDemo {
static void p(String s) {
System.out.println(s);
}
public static void main(String args[ ]) {
File f1 = new File(args[0]);
p("File Name: " + f1.getName());
p("Path: " + f1.getPath());
p("Abs Path: " + f1.getAbsolutePath());
p("Parent: " + f1.getParent());
p(f1.exists() ? "exists" : "does not exist");
p(f1.canWrite() ? "is writeable" : "is not writeable");
```

```
p(f1.canRead() ? "is readable" : "is not readable");
p("is " + (f1.isDirectory() ? "" : "not" + " a directory"));
p(f1.isFile() ? "is normal file" : "might be a named pipe");
p(f1.isAbsolute() ? "is absolute" : "is not absolute");
p("File last modified: " + f1.lastModified());
p("File size: " + f1.length() + " Bytes");
} }
```

13. Write a Java Program to multiply two given matrices.

```
import java.util.*;
class Test {
int r1,c1,r2,c2;
Test(int r1,int c1,int r2,int c2) {
this.r1=r1;
this.c1=c1;
this.r2=r2;
this.c2=c2;
}
int[ ][ ] getArray(int r,int c) {
int arr[][]=new int[r][c];
System.out.println("Enter the elements for "+r+"X"+c+" Matrix:");
Scanner input=new Scanner(System.in);
for(int i=0;i<r;i++)
for(int j=0;j<c;j++)
arr[i][j]=input.nextInt();
return arr;
}
int[ ][ ] findMul(int a[ ][ ],int b[ ][ ]) {
int c[][]=new int[r1][c2];
for (int i=0;i<r1;i++)
for (int j=0;j<c2;j++) {
c[i][j]=0;
for (int k=0;k<r2;k++)
c[i][j]=c[i][j]+a[i][k]*b[k][j];
}

return c;
}
void putArray(int res[ ][ ]) {
System.out.println ("The resultant "+r1+"X"+c2+" Matrix is:");
```

```
for (int i=0;i<r1;i++) {
for (int j=0;j<c2;j++)
System.out.print(res[i][j]+" ");
System.out.println();
} } } //end of Test class
class MatrixMul {
public static void main(String args[ ])throws IOException {
Test obj1=new Test(2,3,3,2);
Test obj2=new Test(2,3,3,2);
int x[ ][ ],y[ ][ ],z[ ][ ];
System.out.println("MATRIX-1:");
x=obj1.getArray(2,3); //to get the matrix from user
System.out.println("MATRIX-2:");
y=obj2.getArray(3,2);
z=obj1.findMul(x,y); //to perform the multiplication
obj1.putArray(z); // to display the resultant matrix
} }
```

14. Write a Java Program that checks whether a given string is a palindrome or not. Ex: MADAM is a palindrome.

```
import java.io.*;
class Palind {
public static void main(String args[ ])throws IOException {
BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
System.out.println("Enter the string to check for palindrome:");
String s1=br.readLine();
StringBuffer sb=new StringBuffer();
sb.append(s1);
sb.reverse();
String s2=sb.toString();
if(s1.equals(s2))
System.out.println("palindrome");
else
System.out.println("not palindrome");
} }
```

15. Write a java program for Method overloading and Constructor overloading.

```
// Method overloading in Java.
public class Sum {
```

```
// Overloaded sum(). This sum takes two int parameters
public int sum(int x, int y)
{
return (x + y);
}
// Overloaded sum(). This sum takes three int parameters
public int sum(int x, int y, int z)
{
return (x + y + z);
}
// Overloaded sum(). This sum takes two double parameters
public double sum(double x, double y)
{
return (x + y);
}
// Driver code
public static void main(String args[])
{
Sum s = new Sum();
System.out.println(s.sum(10, 20));
System.out.println(s.sum(10, 20, 30));
System.out.println(s.sum(10.5, 20.5));
}
}
//Constructor overloading in Java.
class StudentData
{
private int stuID;
private String stuName;
private int stuAge;
StudentData()
{
//Default constructor
stuID = 100;
stuName = "New Student";
stuAge = 18;

StudentData(int num1, String str, int num2)
{
//Parameterized constructor
stuID = num1;
```

```
stuName = str;
stuAge = num2;
}
//Getter and setter methods
public int getStuID() {
return stuID;
}
public void setStuID(int stuID) {
this.stuID = stuID;
}
public String getStuName() {
return stuName;
}
public void setStuName(String stuName) {
this.stuName = stuName;
}
public int getStuAge() {
return stuAge;
}
public void setStuAge(int stuAge) {
this.stuAge = stuAge;
}
public static void main(String args[])
{
//This object creation would call the default constructor
StudentData myobj = new StudentData();
System.out.println("Student Name is: "+myobj.getStuName());
System.out.println("Student Age is: "+myobj.getStuAge());
System.out.println("Student ID is: "+myobj.getStuID());
/*This object creation would call the parameterized constructor StudentData(int,
String, int)*/
StudentData myobj2 = new StudentData(555, "Chaitanya", 25);
System.out.println("Student Name is: "+myobj2.getStuName());
System.out.println("Student Age is: "+myobj2.getStuAge());
System.out.println("Student ID is: "+myobj2.getStuID());
}
}
```

16. Write a java program to display the employee details using Scanner class.

```
import java.util.Scanner;
class Employee
```

```
{
int Id;
String Name;
int Age;
long Salary;
void GetData() // Defining GetData()
{
Scanner sc = new Scanner(System.in);
System.out.print("\n\tEnter Employee Id : ");
Id = Integer.parseInt(sc.nextLine());
System.out.print("\n\tEnter Employee Name : ");
Name = sc.nextLine();
System.out.print("\n\tEnter Employee Age : ");
Age = Integer.parseInt(sc.nextLine());
System.out.print("\n\tEnter Employee Salary : ");
Salary = Integer.parseInt(sc.nextLine());
}
void PutData() // Defining PutData()
{
System.out.print("\n\t" + Id + "\t" +Name + "\t" +Age + "\t" +Salary);
}
public static void main(String args[])
{
Employee[] Emp = new Employee[3];
int i;
for(i=0;i<3;i++)
Emp[i] = new Employee(); // Allocating memory to each object
for(i=0;i<3;i++)
{
System.out.print("\nEnter details of "+ (i+1) +" Employee\n");
Emp[i].GetData();
}
System.out.print("\nDetails of Employees\n");
for(i=0;i<3;i++){

Emp[i].PutData();
}
}
```

17. Write a java program for producer and consumer problem using Threads.

```
// Java program to implement solution of producer consumer problem.
import java.util.LinkedList;
public class Threadexample {
public static void main(String[] args)
throws InterruptedException
{
// Object of a class that has both produce() and consume() methods
final PC pc = new PC();
// Create producer thread
Thread t1 = new Thread(new Runnable() {
@Override
public void run()
{
try {
pc.produce();
}
catch (InterruptedException e) {
e.printStackTrace();
}
}
});
// Create consumer thread
Thread t2 = new Thread(new Runnable() {
@Override
public void run()
{
try {
pc.consume();
}
catch (InterruptedException e) {
e.printStackTrace();
}
}
});
// Start both threads
t1.start();
t2.start();

// t1 finishes before t2
t1.join();
```

```
t2.join();
}
// This class has a list, producer (adds items to list and consumer (removes items).
public static class PC {
// Create a list shared by producer and consumer
// Size of list is 2.
LinkedList<Integer> list = new LinkedList<>();
int capacity = 2;
// Function called by producer thread
public void produce() throws InterruptedException
{
int value = 0;
while (true) {
synchronized (this)
{
// producer thread waits while list is full
while (list.size() == capacity)
wait();
System.out.println("Producer produced-"+ value);
// to insert the jobs in the list
list.add(value++);
// notifies the consumer thread that
// now it can start consuming
notify();
// makes the working of program easier
// to understand
Thread.sleep(1000);
}
}
}
// Function called by consumer thread
public void consume() throws InterruptedException
{
while (true) {
synchronized (this)

{
// consumer thread waits while list
// is empty
while (list.size() == 0)
wait();
```

```
// to retrive the ifrst job in the list
int val = list.removeFirst();
System.out.println("Consumer consumed-" + val);
// Wake up producer thread
notify();
// and sleep
Thread.sleep(1000);
}
}
}
}
}
```

18. Write a Java program that implements a multi-thread application that has three threads.

```
/* The first thread displays "Good Morning" for every one second, the second thread
displays "Hello" for every two seconds and third thread displays "Welcome" for every
three seconds */


class GoodMorning extends Thread {
synchronized public void run() {
try {
int i=0;
while (i<5) {
sleep(1000);
System.out.println("Good morning ");
i++;
}
} catch (Exception e) {
}
}
}
class Hello extends Thread {
synchronized public void run() {
try {
int i=0;
while (i<5) {
sleep(2000);
System.out.println("hello");
i++;
```

```
}
} catch (Exception e) {
}
}
}
class Welcome extends Thread {
synchronized public void run() {
try {
int i=0;
while (i<5) {
sleep(3000);
System.out.println("welcome");
i++;
}
} catch (Exception e) {
}
}
}
class MultithreadDemo {
public static void main(String args[]) {
GoodMorning t1 = new GoodMorning();
Hello t2 = new Hello();

Welcome t3 = new Welcome();
t1.start();
t2.start();
t3.start();
}
}
```

19. Write a Java Program using parameterized constructor with two parameters id and name. While creating the objects obj1 and obj2 passed two arguments so that this constructor gets invoked after creation of obj1 and obj2.

```
class Employee {
intempId;
String empName;
//parameterized constructor with two parameters
Employee(int id, String name){
this.empId = id;
this.empName = name;
```

```
}
void info(){
System.out.println("Id: "+empId+" Name: "+empName);
}
public static void main(String args[]){
Employee obj1 = new Employee(10245,"Chaitanya");
Employee obj2 = new Employee(92232,"Negan");
obj1.info();
obj2.info();
}
}
```

20. Write a java program in which data is read from one file and should be written in another file.name of both file is given through command line arguments.

```
import java.io.*;
classByteCopy
{
public static void main(String arr[])
{
if(arr.length<2)
{
System.out.println("please enter valid array");
System.exit(0);
}
try
{
FileInputStreamfis=new FileInputStream(arr[0]);
FileOutputStreamfos=new FileOutputStream(arr[1]);
long t1=System.currentTimeMillis();
//BufferedReader f= new BufferedReader(new InputStreamReader(new
FileInputStream(arr[0])));
//PrintStreamfos=new PrintStream(new FileOutputStream("k.text"));
while(true)
{
int c=fis.read();
System.out.println("hello");
if(c==-1)
break;
fos.write(c);
System.out.println("hello");
```

```
}
long t2=System.currentTimeMillis();
long t=t2-t1;
fis.close();
fos.close();
System.out.println("operation is complete in" +t+ " seconds");
}catch(Exception ex)
{
//System.out.println("operation is done");
}
}}
```

21. Write a java program in which data is read from one file and should be written in another file line by line.

```
import java.io.*;
class Copy
{
public static void main(String arr[])
{
if(arr.length<2)

{
System.out.println("Usage:Javalinecopy source to target file");
System.exit(0);
}
try
{
FileInputStreamfis= new FileInputStream(arr[0]);
FileOutputStreamfos= new FileOutputStream(arr[1]);
long t1=System.currentTimeMillis();
while(true)
{
String str=fis.readLine();
if(str==null)
break;
fos.println(str);
}
long t2=System.currentTimeMillis();
long t=t2-t1;
fis.close();
fos.close();
System.out.println("successfully copied in" +t+"milliseconds");
```

```
}
catch(Exception ex)
{
System.out.println(ex);
} } }
```

22. Write a program in java that sorts half of element in ascending and rest half of the elements in descending order.

```
import java.util.*;

class sample
{
static void printOrder(int[] a, int n)
  {
int temp;
for(inti=0;i < n-1;i++)
   {
for(int j = 0;j < n/2; j++)
{
      if(a[j]>a[j+1])
        {
temp=a[j];
a[j]=a[j+1];
a[j+1]=temp;
        }
      }

for(int j = n/2;j < n-1; j++)
      {
if(a[j] < a[j+1])
        {
temp=a[j];
a[j]=a[j+1];
a[j+1]=temp;
        }
       }
    }

for(inti = 0; i< n; i++)
System.out.print(a[i] + " ");

  }

public static void main(String[] args)
  {
```

```
int n;
            Scanner sc=new Scanner(System.in);
            System.out.print("enter array size: ");
            n=sc.nextInt();
            int[] array=new int[n];
            System.out.print("enter the elements of array: ");
            for(inti=0;i<n;i++)
            {
                  array[i]=sc.nextInt();
            }

printOrder(array, n);
   }
}
```

23. Write a Java program to remove a specific element from an array

```
importjava.util.Scanner;
importjava.util.Arrays;
importjava.io.BufferedReader;
importjava.io.IOException;
importjava.io.InputStreamReader;

class sample
{
      // Function to remove the element
      public static int[] removeTheElement(int[] arr, int index)
      {
            if (arr == null || index < 0
                  || index >= arr.length) {
                  returnarr;
            }
            int[] anotherArray = new int[arr.length - 1];
            for (inti = 0, k = 0; i<arr.length; i++) {
                  if (i == index)
                  {
                        continue;
                  }
                  anotherArray[k++] = arr[i];
            }
            returnanotherArray;
      }

      // Driver Code
      public static void main(String[] args) throws IOException
      {
```

```
            int n;
            String str1;
            InputStreamReader IN = new InputStreamReader(System.in);
            BufferedReader BR1 = new BufferedReader(IN);
            System.out.print("Please Enter the array size:");
            str1 = BR1.readLine();
            n = Integer.parseInt(str1);
            Scanner scan = new Scanner(System.in);
            int[] arr = new int[n];
            System.out.println("\n\n***Initializing Array***");
            System.out.println("Enter "+ arr.length
               + " integer values:");
            for(inti=0; i<arr.length; i++)
                  {
                        // read input
                        arr[i] = scan.nextInt();
                  }
            System.out.println("***Initialization completed***\n");
            System.out.println("Original Array: "+ Arrays.toString(arr));
            InputStreamReader I = new InputStreamReader(System.in);
            BufferedReader BR = new BufferedReader(I);
            int index;
            String str;
            System.out.print("Please Enter the index no:");
            str = BR.readLine();
            index = Integer.parseInt(str);
            System.out.println("Index to be removed: " + index);
            arr = removeTheElement(arr, index);
            System.out.println("Resultant Array: "+ Arrays.toString(arr));
      }
}
```

24. Write a Java program to remove the duplicate elements of a given array and return the new length of that array.

```
import java.util.*;
import java.util.Scanner;
import java.util.Arrays;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;


public class sample
{
   public static void main(String[] args) throws IOException
```

```
    {
   int n;
         String str1;
         InputStreamReader IN = new InputStreamReader(System.in);
         BufferedReader BR1 = new BufferedReader(IN);

         System.out.print("Please Enter the array size:");
         str1 = BR1.readLine();
         n = Integer.parseInt(str1);



         Scanner scan = new Scanner(System.in);
         int[] arr = new int[n];
         System.out.println("\n\n***Initializing Array***");
         System.out.println("Enter "+ arr.length
             + " integer values:");

         for(int i=0; i < arr.length; i++)
               {
                     // read input
                     arr[i] = scan.nextInt();
               }

         System.out.println("***Initialization completed***\n");
    int[] uniqueArr = new int[arr.length];
    int counter = 0;
    Arrays.sort(arr);
    for (int i = 0; i < arr.length - 1; i++) {
       if (arr[i] != arr[i + 1]) {
          uniqueArr[counter] = arr[i];
          counter++;
       }
    }
    uniqueArr[counter] = arr[arr.length - 1];
    System.out.println("Array with Unique Elements : ");
    for (int i = 0; i <= counter; i++) {
       System.out.println(uniqueArr[i]);
    }
         counter=counter+1;
         System.out.println("the new size of the array is: "+counter);
  }
}
```

25. Write a program in java that handles both 'ArrayIndexOutOfBoundsException' and 'ArithmeticException'.

```
class sample
{
  public static void main(String args[])
  {
    try
    {
      int a=args.length;
      System.out.println("a="+a);
      int b=42/a;
      int c[]={1};
      c[42]=99;
    }
    catch(ArithmeticException e)
    {
      System.out.println("Devide by 0:"+e);
    }
    catch(ArrayIndexOutOfBoundsException e)
    {
     System.out.println("Array index out of bound:"+e);
    }
    System.out.println("After try/catch blocks.");
   }

}
```

26. Write a Java Program to generate Pascal Triangle.

```
          1
        1   1
      1   2   1
    1   3   3   1
  1   4   6   4   1
1   5  10  10   5   1
```

```
public class PascalsTriangle {
  static int factorial(int n) {
    int f;

    for(f = 1; n > 1; n--){
      f *= n;
    }
```

```
    return f;
  }
  static int ncr(int n,int r) {
    return factorial(n) / ( factorial(n-r) * factorial(r) );
  }
  public static void main(String args[]){
    System.out.println();
    int n, i, j;
    n = 5;

    for(i = 0; i <= n; i++) {
      for(j = 0; j <= n-i; j++){
        System.out.print(" ");
      }
      for(j = 0; j <= i; j++){
        System.out.print(" "+ncr(i, j));
      }
      System.out.println();
    }
  }
}
```

27. Consider you are designing vehicles engine with '*speed:int, gear:int*'. You can define your engine functionalities 'speedUp(value)' and 'changeGear(value) in an interface. The class which is implementing the interface should implement all the methods in the interface.

```
import java.util.*;
interface Engine
{
      int gear = 3;
      int speed = 70;
      public void speedUp(int s);
      public void changeGear(int g);
}
class vehicle implements Engine
{

      public void speedUp(int s)
      {
```

```
            intch= s-speed;
            System.out.println("the speed changed from "+speed+" to "+s);
            System.out.println("the speed up of the car is "+ch);
      }
      public void changeGear(int g)
      {
            int ch1= g-gear;
            System.out.println("the gear changed from "+gear+" to "+g);
            System.out.println("the change in gear is "+ch1);
      }
}
class Vehicle_Engine
{

      public static void main (String args[])
      {
            Engine e= new vehicle();
            e.speedUp(100);
            e.changeGear(5);
      }
}
```

28. Create a base class 'Square' having instance variable side:double. Initiate variable using constructor, a method 'getVolume() : double' that calculates volume and print it. Create a derived class 'Cylinder' having instance variable height:double. Initiate variables of both classes through constructor, override method 'getVolume() : double' to calculate volume of cylinder taking 'side' variable of base class as 'radius' and print it.

```
import java.util.*;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
class square
{
   double side;
   double vol;
   square(double side)
   {
      this.side=side;
```

```
    }
    public void getVolume()
    {
      vol=java.lang.Math.pow(side,3);
      System.out.println("The volume of square is:"+vol);
    }
}
class cylinder extends square
{   double height;
    cylinder(double side,double height)
    {
       super(side);
       this.height=height;
    }




    public void getVolume()
    {
      vol=(3.14*java.lang.Math.pow(side,2)*height);
      System.out.println("The volume of Cylinder is:"+vol);
    }


}
public class sample{
    public static void main(String[] args) throws IOException
    {
                double side,height;
                String str1,str2;
                InputStreamReader IN = new InputStreamReader(System.in);
                BufferedReader BR1 = new BufferedReader(IN);

                System.out.print("Please Enter the side:");
                str1 = BR1.readLine();
                side = Double.parseDouble(str1);
                System.out.print("Please Enter the height:");
                str2 = BR1.readLine();
                height = Double.parseDouble(str1);
```

```
        square s=new square(side);
        s.getVolume();
        cylinder c=new cylinder(side,height);
        c.getVolume();

    }
}
```

29. A class called MyPoint, which models a 2D point with x and y coordinates. It contains:

• Two instance variables x (int) and y (int).

• A default (or "no-argument" or "no-arg") constructor that construct a point at the default location of (0, 0).

• A overloaded constructor that constructs a point with the given x and y coordinates.

• A method setXY() to set both x and y.

• A method getXY() which returns the x and y in a 2-element int array.

• A toString() method that returns a string description of the instance in the format "(*x*, *y*)".

• A method called distance(int x, int y) that returns the distance from *this* point to another point at the given (x, y) coordinates, Write the MyPoint class. Also write a test driver (called TestMyPoint) to test all the public methods defined in the class.

```
public class Mypoint
{
int x;
int y;

int[] a=new int[2];
Mypoint()
    {
        x=0;
        y=0;
    }
Mypoint(int x,int y)
    {
        this.x=x;
        this.y=y;
    }
```

```
public void SetXY(int x,int y)
   {
       this.x=x;
       this.y=y;
   }
public int[] getXY()
   {
a[0]=x;
a[1]=y;
return a;

   }
public String toString()
   {
      String s="("+x+" ,"+y+")";
return s;
   }
public int distance()
   {
int z;
      z=y-x;
return z;


   }

public static void main(String[] args)
   {
      Mypoint m = new  Mypoint();
      Mypoint m1 = new  Mypoint(6,9);
m.SetXY(3,5);
int b[]=m.getXY();
System.out.println("Value of x and y is:"+b[0]+" and "+ b[1]);
      String t=m.toString();
System.out.println(t);
int z=m.distance();
System.out.println("The distance between x and y is:"+z);

  }
}
```

30. Star Patterns in Java:

a) Pyramid Program

```
   *
  * *
 * * *
* * * *
* * * * *
```

```
public class sample
{
   public static void pyramidPattern(int n)
   {
      for (int i=0; i<n; i++) //outer loop for number of rows(n)
            {
        for (int j=n-i; j>1; j--) //inner loop for spaces
         {
            System.out.print(" "); //print space
         }
         for (int j=0; j<=i; j++ ) //inner loop for number of columns
         {
            System.out.print("* "); //print star
         }

         System.out.println(); //ending line after each row
      }
   }

   public static void main(String args[]) //driver function
   {
      int n = 5;
      pyramidPattern(n);
   }
}
```

b) Right Triangle Star Pattern

```
*
* *
* * *
* * * *
* * * * *
```

```
public class Sample
{
   public static void rightTriangle(int n)
   {
      int i, j;
      for(i=0; i<n; i++) //outer loop for number of rows(n)
      { for(j=2*(n-i); j>=0; j--) // inner loop for spaces
         {
            System.out.print(" "); // printing space
         }
         for(j=0; j<=i; j++) //  inner loop for columns
         {
            System.out.print("* "); // print star
         }
         System.out.println(); // ending line after each row
      }
   }
   public static void main(String args[])
   {
      int n = 5;
      rightTriangle(n);
   }
}
```

c) Left Triangle Star Pattern

```
            *
         *  *
      *  *  *
   *  *  *  *
*  *  *  *  *
```

```
public class sample
{
   public static void printStars(int n)
   {
      int i, j;
      for(i=0; i<n; i++) //outer loop for number of rows(n) { for(j=2*(n-i); j>=0; j--)
// inner loop for spaces
         {
            System.out.print(" "); // printing space
         }
         for(j=0; j<=i; j++) //  inner loop for columns
         {
```

```
            System.out.print("* "); // print star
         }
         System.out.println(); // ending line after each row
      }
   }
   public static void main(String args[])
   {
      int n = 5;
      printStars(n);
   }
}
```

d) Diamond Shape Pattern Program in Java

Enter the number of rows: 5

```
    *
   ***
  *****
 *******
*********
 *******
  *****
   ***
    *
```

```
import java.util.Scanner;
public class sample
{
public static void main(String args[])
{
int n, i, j, space = 1;
System.out.print("Enter the number of rows: ");
Scanner s = new Scanner(System.in);
n = s.nextInt();
space = n - 1;
for (j = 1; j<= n; j++)
{
for (i = 1; i<= space; i++)
{
System.out.print(" ");
}
space--;
for (i = 1; i <= 2 * j - 1; i++)
{
```

```
System.out.print("*");
}
System.out.println("");
}
space = 1;
for (j = 1; j<= n - 1; j++)
{
for (i = 1; i<= space; i++)
{
System.out.print(" ");
}
space++;
for (i = 1; i<= 2 * (n - j) - 1; i++)
{
System.out.print("*");
}
System.out.println("");
}
}
}
```

e) Downward Triangle Star Pattern

Enter the number of rows: 5

```
* * * * *
* * * *
* * *
* *
*
```

```
import java.util.Scanner;
public class sample
{
   public static void main(String[] args)
   {
   Scanner sc = new Scanner(System.in);

   System.out.println("Enter the number of rows: "); //takes input from user

   int rows = sc.nextInt();

   for (int i= rows-1; i>=0 ; i--)
   {
   for (int j=0; j<=i; j++)
   {
   System.out.print("*" + " ");
```

```
    }
    System.out.println();
    }
    sc.close();
    }
    }
```

f) Reversed Pyramid Star Pattern

Enter the number of rows: 5

```
 * * * * *
  * * * *
   * * *
    * *
     *
```

```
import java.util.Scanner;
public class sample
{
   public static void main(String[] args)
{
   Scanner sc = new Scanner(System.in);
   System.out.println("Enter the number of rows: ");

   int rows = sc.nextInt();
   for (int i= 0; i<= rows-1 ; i++)
   {
      for (int j=0; j<=i; j++)
      {
         System.out.print(" ");
      }
      for (int k=0; k<=rows-1-i; k++)
      {
         System.out.print("*" + " ");
      }
      System.out.println();
   }
   sc.close();

}
```

```
}
```

g) Right Pascal's Triangle

Enter the number of rows: 5

```
*
* *
* * *
* * * *
* * * * *
* * * *
* * *
* *
*
```

```
import java.util.Scanner;
public class sample
{
   public static void main(String[] args)
   {
      Scanner sc = new Scanner(System.in);
      System.out.println("Enter the number of rows: ");

      int rows = sc.nextInt();
      for (int i= 0; i<= rows-1 ; i++)
      {
         for (int j=0; j<=i; j++) { System.out.print("*"+ " "); }
System.out.println(""); } for (int i=rows-1; i>=0; i--)
      {
         for(int j=0; j <= i-1;j++)
         {
            System.out.print("*"+ " ");
         }
         System.out.println("");
      }
      sc.close();
   }
}
```

h) Sandglass Star Pattern

Enter the number of rows: 5

```
* * * * *
 * * * *
  * * *
   * *
    *
    *
   * *
  * * *
 * * * *
* * * * *
```

```
import java.util.Scanner;
public class Edureka
{
   public static void main(String[] args)
   {
      Scanner sc = new Scanner(System.in);
      System.out.println("Enter the number of rows: ");

      int rows = sc.nextInt();
      for (int i= 0; i<= rows-1 ; i++)
      {
         for (int j=0; j <i; j++)
         {
            System.out.print(" ");
         }
         for (int k=i; k<=rows-1; k++) { System.out.print("*" + " "); }
System.out.println(""); } for (int i= rows-1; i>= 0; i--)
      {
         for (int j=0; j< i ;j++)
         {
            System.out.print(" ");
         }
         for (int k=i; k<=rows-1; k++)
         {
            System.out.print("*" + " ");
         }
         System.out.println("");
      }
      sc.close();
   }
}
```

i) Triangle Star pattern

Enter the number of rows: 5

```
    *
   * *
  *   *
 *     *
*********
```

```
import java.util.Scanner;
public class sample
{
    public static void main(String[] args)
      {
         Scanner sc = new Scanner(System.in);

         System.out.println("Enter the number of rows: ");

         int rows = sc.nextInt();

         for (int i=1; i<= rows ; i++)
         {
            for (int j = i; j < rows ; j++) {
               System.out.print(" ");
            }
            for (int k = 1; k <= (2*i -1) ;k++) {
               if( k==1 || i == rows || k==(2*i-1)) {
                  System.out.print("*");
               }
               else {
                  System.out.print(" ");
               }
            }
            System.out.println("");
         }
         sc.close();
      }
   }
```

j) Diamond Star Pattern

Enter the number of rows: 5

```
    *
   * *
  *   *
 *     *
*       *
 *     *
  *   *
   * *
    *
```

```
import java.util.Scanner;
public class sample
{
   public static void main(String[] args)
{
   Scanner sc = new Scanner(System.in);

   System.out.println("Enter the number of rows: ");

   int rows = sc.nextInt();
   for (int i=1; i<= rows ; i++) { for (int j = rows; j > i ; j--) {
         System.out.print(" ");
      }
      System.out.print("*");
      for (int k = 1; k < 2*(i -1) ;k++) { System.out.print(" "); }
if( i==1)
{ System.out.println(""); }
else
{ System.out.println("*"); } }
for (int i=rows-1; i>= 1 ; i--)
      {
      for (int j = rows; j > i ; j--) {
         System.out.print(" ");
      }
      System.out.print("*");
      for (int k = 1; k < 2*(i -1) ;k++) {
         System.out.print(" ");
      }
```

```
        if( i==1)
            System.out.println("");
        else
            System.out.println("*");
    }
    sc.close();
}
}
```

k) Diamond Pattern Program in Java

```
   1
  212
 32123
4321234
 32123
  212
   1
```

```
import java.util.Scanner;

public class sample
{
    public static void main(String[] args) {
        for (int i = 1; i <= 4; i++)
        {
            int n = 4;

            for (int j = 1; j<= n - i; j++) { System.out.print(" "); }
             for (int k = i; k >= 1; k--)
            {
                System.out.print(k);
            }
            for (int l = 2; l <= i; l++) { System.out.print(l); } System.out.println(); }
         for (int i = 3; i >= 1; i--)
        {
            int n = 3;

            for (int j = 0; j<= n - i; j++) { System.out.print(" "); }
         for (int k = i; k >= 1; k--)
            {
                System.out.print(k);
            }
```

```
            for (int l = 2; l <= i; l++)
            {
               System.out.print(l);
            }

            System.out.println();
         }

      }
}
```

31. Write a program in java that accepts a 2D matrix and prints the matrix with row minimum and column minimum values.

```
importjava.util.Scanner;
public class sample
{
	static void smallestInRow(int mat[][], int n, int m) {
		System.out.print(" { ");
		for (int i = 0; i < n; i++) {
			int minm = mat[i][0];
			for (int j = 1; j < m; j++) {
				if (mat[i][j] < minm) {
					minm = mat[i][j];
				}
			}
			System.out.print(minm + ", ");
		}
		System.out.println("}");
	}
	static void smallestInCol(int mat[][], int n, int m) {
		System.out.print(" { ");
		for (int i = 0; i < m; i++) {
			int minm = mat[0][i];
			for (int j = 1; j < n; j++) {
				if (mat[j][i] < minm) {
					minm = mat[j][i];
				}
			}
			System.out.print(minm + ", ");
		}
		System.out.print("}");
	}

// Driver code
	public static void main(String args[]) {
```

```
            int m, n, i, j;
        Scanner in = null;

in = new Scanner(System.in);
System.out.println("Enter the number " + "of rows of the matrix");
        m = in.nextInt();
System.out.println("Enter the number "
                           + "of columns of the matrix");
        n = in.nextInt();

        // Declare the matrix
int mat[][] = new int[m][n];

        // Read the matrix values
System.out.println("Enter the elements of the matrix");
for (i = 0; i < m; i++)
for (j = 0; j < n; j++)
mat[i][j] = in.nextInt();
            System.out.print("Minimum element of each row is ");
            smallestInRow(mat, m, n);
            System.out.print("\nMinimum element of each column is ");
            smallestInCol(mat, m, n);
        }
}
```

32. Write a program in java to delete all consonants from an input string and print the result string.

```
import java.util.Arrays;
import java.util.List;
public class RemoveConsonantsFromString
{
  public static void main(String[] args)
  {
    String str = "hello world core java";
    System.out.println("Remove consonants from a string: ");
    System.out.println(removeConsonantsFunction(str));
  }
  static boolean checkAlphabet(char ch)
  {
    if(ch >= 'a' && ch <= 'z')
      return true;
    if(ch >= 'A' && ch <= 'Z')
      return true;
    return false;
  }
```

```
  static String removeConsonantsFunction(String strConsonant)
  {
    Character[] chVowel = { 'a', 'e', 'i', 'o', 'u', 'A', 'E', 'I', 'O', 'U' };
    List<Character> li = Arrays.asList(chVowel);
    StringBuffer sb = new StringBuffer(strConsonant);
    for(int a = 0; a < sb.length(); a++)
    {
      if(checkAlphabet(sb.charAt(a)) && !li.contains(sb.charAt(a)))
      {
        sb.replace(a, a + 1, "");
        a--;
      }
    }
    return sb.toString();
  }
}
```

33. Write a Java program to remove a specific element from an array.

```
import java.util.Scanner;
public class Delete
{
   public static void main(String[] args)
   {
      int n, x, flag = 1, loc = 0;
      Scanner s = new Scanner(System.in);
      System.out.print("Enter no. of elements you want in array:");
      n = s.nextInt();
      int a[] = new int[n];
      System.out.println("Enter all the elements:");
      for (int i = 0; i < n; i++)
      {
         a[i] = s.nextInt();
      }
      System.out.print("Enter the element you want to delete:");
      x = s.nextInt();
      for (int i = 0; i < n; i++)
      {
         if(a[i] == x)
         {
            flag =1;
            loc = i;
            break;
         }
         else
         {
```

```
            flag = 0;
         }
      }
      if(flag == 1)
      {
         for(int i = loc+1; i < n; i++)
         {
            a[i-1] = a[i];
         }
         System.out.print("After Deleting:");
         for (int i = 0; i < n-2; i++)
         {
            System.out.print(a[i]+",");
         }
         System.out.print(a[n-2]);
      }
      else
      {
         System.out.println("Element not found");
      }
   }
}
```

34. Write a Java program to insert an element (specific position) into an array.

```
import java.util.Scanner;
public class Insert_Array
{
   public static void main(String[] args)
   {
      int n, pos, x;
      Scanner s = new Scanner(System.in);
      System.out.print("Enter no. of elements you want in array:");
      n = s.nextInt();
      int a[] = new int[n+1];
      System.out.println("Enter all the elements:");
      for(int i = 0; i < n; i++)
      {
         a[i] = s.nextInt();
      }
      System.out.print("Enter the position where you want to insert element:");
      pos = s.nextInt();
      System.out.print("Enter the element you want to insert:");
      x = s.nextInt();
      for(int i = (n-1); i >= (pos-1); i--)
      {
```

```
            a[i+1] = a[i];
        }
        a[pos-1] = x;
        System.out.print("After inserting:");
        for(int i = 0; i < n; i++)
        {
            System.out.print(a[i]+",");
        }
        System.out.print(a[n]);
    }
}
```

35. Write a Java program to find all pairs of elements in an array whose sum is equal to a specified number.

```
import java.util.Arrays;
import java.util.Scanner;
public class sample {
   public static void main(String args[]){
      //Reading the array from the user
      Scanner sc = new Scanner(System.in);
      System.out.println("Enter the size of the array that is to be created: ");
      int size = sc.nextInt();
      int[] myArray = new int[size];
      System.out.println("Enter the elements of the array: ");
      for(int i=0; i<size; i++){
         myArray[i] = sc.nextInt();
      }
      //Reading the number
      System.out.println("Enter the number: ");
      int num = sc.nextInt();
      System.out.println("The array created is: "+Arrays.toString(myArray));
      System.out.println("indices of the elements whose sum is: "+num);
      for(int i=0; i<myArray.length; i++){
         for (int j=i; j<myArray.length; j++){
            if((myArray[i]+myArray[j])== num && i!=j){
               System.out.println(i+", "+j);
            }
         }
      }
   }
}
```

36. Write a Java program to remove the duplicate elements of a given array and return the new length of that array.

a) Remove Duplicate Element in Array using Temporary Array

```
public class RemoveDuplicateInArrayExample{
public static int removeDuplicateElements(int arr[], int n){
        if (n==0 || n==1){
            return n;
        }
        int[] temp = new int[n];
        int j = 0;
        for (int i=0; i<n-1; i++){
            if (arr[i] != arr[i+1]){
                temp[j++] = arr[i];
            }
         }
        temp[j++] = arr[n-1];
        // Changing original array
        for (int i=0; i<j; i++){
            arr[i] = temp[i];
        }
        return j;
    }

    public static void main (String[] args) {
        int arr[] = {10,20,20,30,30,40,50,50};
        int length = arr.length;
        length = removeDuplicateElements(arr, length);
        //printing array elements
        for (int i=0; i<length; i++)
          System.out.print(arr[i]+" ");
    }
}
```

b) Remove Duplicate Element in Array using separate index

```
public class RemoveDuplicateInArrayExample2{
    public static int removeDuplicateElements(int arr[], int n){
        if (n==0 || n==1){
            return n;
        }
        int j = 0;//for next element
        for (int i=0; i < n-1; i++){
            if (arr[i] != arr[i+1]){
                arr[j++] = arr[i];
            }
        }
```

```
        arr[j++] = arr[n-1];
        return j;
    }

    public static void main (String[] args) {
        int arr[] = {10,20,20,30,30,40,50,50};
        int length = arr.length;
        length = removeDuplicateElements(arr, length);
        //printing array elements
        for (int i=0; i<length; i++)
            System.out.print(arr[i]+" ");
    }
}
```

37. Write a Java program to find the length of the longest consecutive elements sequence from a given unsorted array of integers.
Sample array: [49, 1, 3, 200, 2, 4, 70, 5]
The longest consecutive elements sequence is [1, 2, 3, 4, 5], therefore the program will return its length 5.

```
import java.util.HashSet;
public class Sample {
   public static void main(String[] args) {
        int nums[] = {49, 1, 3, 200, 2, 4, 70, 5};
                System.out.println("Original array length: "+nums.length);
                System.out.print("Array elements are: ");
       for (int i = 0; i < nums.length; i++)
        {
            System.out.print(nums[i]+" ");
        }
                System.out.println("\nThe new length of the array is:
"+longest_sequence(nums));

    }

    public static int longest_sequence(int[] nums) {
      final HashSet<Integer> h_set = new HashSet<Integer>();
        for (int i : nums) h_set.add(i);

        int longest_sequence_len = 0;
        for (int i : nums) {
            int length = 1;
            for (int j = i - 1; h_set.contains(j); --j) {
                h_set.remove(j);
                ++length;
            }
```

```
            for (int j = i + 1; h_set.contains(j); ++j) {
                h_set.remove(j);
                ++length;
            }
            longest_sequence_len = Math.max(longest_sequence_len, length);
        }
        return longest_sequence_len;
    }
}
```

38. Write a java program to compare two strings lexicographically.

```
public class Sample {
public static void main(String[] args)
    {
        String str1 = "This is Exercise 1";
        String str2 = "This is Exercise 2";

        System.out.println("String 1: " + str1);
        System.out.println("String 2: " + str2);

        // Compare the two strings.
        int result = str1.compareTo(str2);

        // Display the results of the comparison.
        if (result < 0)
        {
            System.out.println("\"" + str1 + "\"" +
                " is less than " +
                "\"" + str2 + "\"");
        }
        else if (result == 0)
        {
            System.out.println("\"" + str1 + "\"" + " is equal to " +
                "\"" + str2 + "\"");
        }
        else // if (result > 0)
        {
            System.out.println("\"" + str1 + "\"" + " is greater than " +
                "\"" + str2 + "\"");
        }
    }
}
```

39. Write a Java program to find whether a region in the current string matches a region in another string.
Sample Output:
str1[0 - 7] == str2[28 - 35]? true
str1[9 - 15] == str2[9 - 15]? false

```
public class Sample {

  public static void main(String[] args)
    {
      //String str1 = "Red Green Orange Yellow";
       //String str2 = "Yellow Orange Green Red";

       String str1 = "Shanghai Houston Colorado Alexandria";
       String str2 = "Alexandria Colorado Houston Shanghai";

       // Determine whether characters 0 through 7 in str1
       // match characters 28 through 35 in str2.
       boolean match1 = str1.regionMatches(0, str2, 28, 8);

       // Determine whether characters 9 through 15 in str1
       // match characters 9 through 15 in str2.
       boolean match2 = str1.regionMatches(9, str2, 9, 8);

       // Display the results of the regionMatches method calls.
       System.out.println("str1[0 - 7] == str2[28 - 35]? " + match1);
       System.out.println("str1[9 - 15] == str2[9 - 15]? " + match2);
    }
}
```

40. Write a Java program to print all permutations of a given string with repetition.
Sample Output:
The given string is: PQR
The permuted strings are:
PPP, PPQ, PPR, RRP, RRQ, RRR

```
import java.util.*;
public class Main {

 public static void main(String[] args) {
  permutationWithRepeation("PQR");

 }
```

```
 private static void permutationWithRepeation(String str1) {
  System.out.println("The given string is: PQR");
  System.out.println("The permuted strings are:");
  showPermutation(str1, "");
 }

 private static void showPermutation(String str1, String NewStringToPrint) {

  if (NewStringToPrint.length() == str1.length()) {
   System.out.println(NewStringToPrint);
   return;
  }
  for (int i = 0; i < str1.length(); i++) {

   showPermutation(str1, NewStringToPrint + str1.charAt(i));
  }
 }
}
```

41. Write a Java method to count all words in a string.

```
import java.util.Scanner;
public class Exercise5 {

  public static void main(String[] args)
    {
        Scanner in = new Scanner(System.in);
        System.out.print("Input the string: ");
        String str = in.nextLine();

        System.out.print("Number of words in the string: " + count_Words(str)+"\n");
    }

 public static int count_Words(String str)
    {
       int count = 0;
        if (!(" ".equals(str.substring(0, 1))) || !(" ".equals(str.substring(str.length() -
1))))
        {
            for (int i = 0; i < str.length(); i++)
            {
                if (str.charAt(i) == ' ')
                {
                    count++;
                }
            }
```

```
            count = count + 1;
        }
        return count; // returns 0 if string starts or ends with space " ".
    }
}
```

42. Write a program in java to create Box class with parameterized constructor with an object argument to initialize length, breadth and height also create a function volume which returns the volume of the box and print it in main method.

```
class Box {
  double width;
  double height;
  double depth;

  // This is the constructor for Box.
  Box(double w, double h, double d) {
    width = w;
    height = h;
    depth = d;
  }

  // compute and return volume
  double volume() {
    return width * height * depth;
  }
}

class BoxDemo {
  public static void main(String args[]) {
    // declare, allocate, and initialize Box objects
    Box mybox1 = new Box(10, 20, 15);
    Box mybox2 = new Box(3, 6, 9);

    double vol;

    // get volume of first box
    vol = mybox1.volume();
    System.out.println("Volume is " + vol);

    // get volume of second box
    vol = mybox2.volume();
    System.out.println("Volume is " + vol);
  }
}
```

43. Write a Java program to search an element using binary search.

```
import java.util.Scanner;
public class JavaProgram
{
 public static void main(String args[])
 {
 int n, i, search, first, last, middle;
 int arr[] = new int[50];
 Scanner scan = new Scanner(System.in);

 System.out.print("Enter Total Number of Elements : ");
 n = scan.nextInt();

 System.out.print("Enter " +n+ " Elements : ");
 for(i=0; i<n; i++)
 {
 arr[i] = scan.nextInt();
 }

 System.out.print("Enter a Number to Search..");
 search = scan.nextInt();

 first = 0;
 last = n-1;
 middle = (first+last)/2;

 while(first <= last)
 {
 if(arr[middle] < search)
 {
 first = middle+1;
 }
 else if(arr[middle] == search)
 {
System.out.print(search+ " Found at Location "
+middle);
 break;
 }
 else
 {
 last = middle - 1;
 }
 middle = (first+last)/2;
 }
 if(first > last)
 {
```

```
System.out.print("Not Found..!! " +search+ " is not
Present in the List.");
 }
 }
}
```

44. Write a Java program to count the frequency of words, characters in the given line of text.

```
import java.util.Scanner;
public class Frequency_of_Characters_in_String
{
 public static void main(String args[])
 {
 String str;
 Scanner scan = new Scanner(System.in);

 System.out.print("Enter a String : ");
 str=scan.nextLine();

System.out.print("Total Number of Words in Entered Sentence is
" + countWords(str)+"\n\n");
System.out.print("Total Number of Characters in Entered
Sentence is " + countCharacter(str));
 }

public static int countCharacter(String str)
{
 int i, j, k, count=0;
 char c, ch;
 i=str.length();
 for(c='A'; c<='z'; c++)
 {
 k=0;
 for(j=0; j<i; j++)
 {
 ch = str.charAt(j);
 if(ch == c)
 {
 k++;
 count++;
 }
 }
 if(k>0)
 {
System.out.println("The character " + c + " has occurred for
```

```
" + k + " times");
 }
 }
 return count;
 }
public static int countWords(String str)
{
 int count = 1;
 for(int i=0; i<=str.length()-1; i++)
 {
 if(str.charAt(i) == ' ' && str.charAt(i+1)!=' ')
 {
 count++;
 }
 }
 return count;
 }
}
```

45. Write a Java Program to implement multilevel inheritance by applying various access controls to its data members and methods.

```
import java.io.DataInputStream;
class Student
{
private int rollno;
private String name;
DataInputStream dis=new DataInputStream(System.in);
public void getrollno()
{
try
{
System.out.println("Enter rollno ");
rollno=Integer.parseInt(dis.readLine());
System.out.println("Enter name ");
name=dis.readLine();
}
catch(Exception e){ }
}
void putrollno()
{
System.out.println("Roll No ="+rollno);
System.out.println("Name ="+name);
}
}
class Marks extends Student
```

```
{
protected int m1,m2,m3;
void getmarks()
{
try
{
System.out.println("Enter marks :");
m1=Integer.parseInt(dis.readLine());
m2=Integer.parseInt(dis.readLine());
m3=Integer.parseInt(dis.readLine());
}
catch(Exception e) { }
}
void putmarks()
{
System.out.println("m1="+m1);
System.out.println("m2="+m2);
System.out.println("m3="+m3);
}
}
class Result extends Marks
{
private float total;
void compute_display()
{
total=m1+m2+m3;
System.out.println("Total marks :" +total);
}
}
class MultilevelDemo
{
public static void main(String arg[])
{
Result r=new Result();
r.getrollno();
r.getmarks();
r.putrollno();
r.putmarks();
r.compute_display();
}
}
```

46. Write a Java Program to implement Vector class and its methods.

```
import java.lang.*;
import java.util.Vector;
import java.util.Enumeration;
```

```
class vectordemo
{
public static void main(String arg[])
{
Vector v=new Vector();
v.addElement("one");
v.addElement("two");
v.addElement("three");
v.insertElementAt("zero",0);
v.insertElementAt("oops",3);
v.insertElementAt("four",5);
System.out.println("Vector Size :"+v.size());
System.out.println("Vector apacity :"+v.capacity());
System.out.println(" The elements of a vector are :");
Enumeration e=v.elements();
while(e.hasMoreElements())
System.out.println(e.nextElement() +" ");
System.out.println();
System.out.println("The first element is : " +v.firstElement());
System.out.println("The last element is : " +v.lastElement());
System.out.println("The object oops is found at position :
"+v.indexOf("oops"));
v.removeElement("oops");
v.removeElementAt(1);
System.out.println("After removing 2 elements ");
System.out.println("Vector Size :"+v.size());
System.out.println("The elements of vector are :");
for(int i=0;i<v.size();i++)
System.out.println(v.elementAt(i)+" ");
}
}
```

47. Write a program to demonstrate use of user defined packages.

Package is a keyword of Java followed by the package name. Just writing the package statement followed by the name creates a new package; see how simple Java is to practice. For this reason, Java is known as a production language.

While create User Defined Packages Java, the order of statements is very important. The order must be like this, else, compilation error.

1. Package statement
2. Import statement
3. Class declaration

```
package forest;
import java.util.*;
public class Tiger
{
 public void getDetails(String nickName, int weight)
 {
System.out.println("Tiger nick name is " + nickName);
 System.out.println("Tiger weight is " + weight);
 }
}
```

When the code is ready, the next job is compilation. We must compile with package notation. Package notation uses –d compiler option as follows.

C:\snr > javac -d . Tiger.java

The –d compiler option creates a new folder called forest and places the Tiger.class in it. The dot (.) is an operating system's environment variable that indicates the current directory. It is an instruction to the OS to create a directory called forest and place the Tiger.class in it.
Using User Defined Packages Java

After creating the package let us use it.

2nd step: Set the classpath from the target directory.
Let us assume D:\sumathi is the target directory. Let us access Tiger.class in forest package from here.

From the target directory set the classpath following way.
D:\vinay> set classpath=C:\snr;%classpath%;

classpath is another environment variable which gives the address of the forest directory to the OS. %classpath% informs the OS to append the already existing classpath to the current classpath that is right now set.

3rd Step: Now finally, write a program from the target directory D:\vinay and access the package.
D:\vinay> notepad Animal.java

```
import forest.Tiger;
public class Animal
{
 public static void main(String args[])
 {
 Tiger t1 = new Tiger ();
```

```
 t1.getDetails("Everest", 50);
 }
}
```

The compilation and execution is as usual as follows.
D:\vinay>javac Animal.java
D:\vinay> java Animal

48. Write a java program to implement exception handling using multiple catch statements.

```
public class MultipleCatchBlock1
{
public static void main(String[] args)
{
 try
{
 int a[]=new int[5];
 a[5]=30/0;
 }
 catch(ArithmeticException e)
 {
System.out.println("Arithmetic Exception
occurs");
 }
 catch(ArrayIndexOutOfBoundsException e)
 {
System.out.println("ArrayIndexOutOfBounds Exc
eption occurs");
 }
 catch(Exception e)
 {
 System.out.println("Parent Exception occurs");
 }
 System.out.println("rest of the code");
 }
}
```

49. Design stack class with necessary exception handling.

```
import java.util.*;
/* Class arrayStack */
class arrayStack
{
protected int arr[];
protected int top, size, len;
```

```
/* Constructor for arrayStack */
public arrayStack(int n)
{
size = n;
len = 0;
arr = new int[size];
top = -1;
}
/* Function to check if stack is empty */
public boolean isEmpty()
{
return top == -1;
}
/* Function to check if stack is full */
public boolean isFull()
{
return top == size -1 ;
}
/* Function to get the size of the stack */
public int getSize()
{
return len ;
}
/* Function to check the top element of the stack */
public int peek()
{
if( isEmpty() )
throw new NoSuchElementException("Underflow
Exception");
return arr[top];
}
/* Function to add an element to the stack */
public void push(int i)
{
if(top + 1 >= size)
throw new IndexOutOfBoundsException("Overflow
Exception");
if(top + 1 < size )
arr[++top] = i;
len++ ;
}
/* Function to delete an element from the stack */
public int pop()
{
if( isEmpty() )
throw new NoSuchElementException("Underflow
```

```
Exception");
len-- ;
return arr[top--];
}
/* Function to display the status of the stack */
public void display()
{
System.out.print("\nStack = ");
if (len == 0)
{
System.out.print("Empty\n");
return ;
}
for (int i = top; i >= 0; i--)
System.out.print(arr[i]+" ");
System.out.println();
}
}
/* Class StackImplement */
public class StackImplement
{
public static void main(String[] args)
{
Scanner scan = new Scanner(System.in);
System.out.println("Stack Test\n");
System.out.println("Enter Size of Integer Stack ");
int n = scan.nextInt();
/* Creating object of class arrayStack */
arrayStack stk = new arrayStack(n);
/* Perform Stack Operations */
char ch;
do
{
System.out.println("\nStack Operations");
System.out.println("1. push");
System.out.println("2. pop");
System.out.println("3. peek");
System.out.println("4. check empty");
System.out.println("5. check full");
System.out.println("6. size");
int choice = scan.nextInt();
switch (choice)
{
case 1 : System.out.println("Enter integer
element to push");
try
```

```
{
stk.push( scan.nextInt() );
}
catch (Exception e)
{
System.out.println("Error : " +
e.getMessage());
}
break;
case 2 :
try
{
System.out.println("Popped Element = " +
stk.pop());
}
catch (Exception e)
{
System.out.println("Error : " +
e.getMessage());
}
break;
case 3 :
try
{
System.out.println("Peek Element = " +
stk.peek());
}
catch (Exception e)
{
System.out.println("Error : " +
e.getMessage());
}
break;
case 4 :
System.out.println("Empty status = " +
stk.isEmpty());
break;
case 5 :
System.out.println("Full status = " +
stk.isFull());
break;
case 6 :
System.out.println("Size = " +
stk.getSize());
break;
default :
```

```
 System.out.println("Wrong Entry \n ");
break;
}
/* display stack */
stk.display();
System.out.println("\nDo you want to continue (Type y
or n) \n");
ch = scan.next().charAt(0);
}
while (ch == 'Y'|| ch == 'y');
}
}
```

50. Illustrate creation of thread by extending Thread class.

```
import java.lang.Thread;
class A extends Thread
{
public void run()
{
System.out.println("thread A is started:");
for(int i=1;i<=5;i++)
{
System.out.println("\t from thread A:i="+i);
}
System.out.println("exit from thread A:");
}
}
class B extends Thread
{
public void run()
{
System.out.println("thread B is started:");
for(int j=1;j<=5;j++)
{
System.out.println("\t from thread B:j="+j);
}
System.out.println("exit from thread B:");
}
}
class C extends Thread
{
public void run()
{
System.out.println("thread C is started:");
for(int k=1;k<=5;k++)
{
```

```
System.out.println("\t from thread C:k="+k);
}
System.out.println("exit from thread C:");
}
}
class Threadtest
{
public static void main(String arg[])
{
new A().start();
new B().start();
new C().start();
}
}
```

51. Write a Java Program to check whether a number is perfect or not.

```
import java.util.Scanner;
public class PerfectNumber
{
public static void main(String args[])
{
long n, sum=0;
Scanner sc=new Scanner(System.in);
System.out.print("Enter the number: ");
n=sc.nextLong();
int i=1;
//executes until the condition becomes false
while(i <= n/2)
{
if(n % i == 0)
{
//calculates the sum of factors
Sum = sum + i;
} //end of if
//after each iteration, increments the value of variable i by 1
i++;
} //end of while
//compares sum with the number
if(sum==n)
{
//prints if sum and n are equal
System.out.println(n+" is a perfect number.");
} //end of if
else
//prints if sum and n are not equal
```

```
System.out.println(n+" is not a perfect number.");
}
}
```

52. Write a Java Program to check whether a number is peterson or not.

```
/checks whether a number entered by user is peterson number or not
import java.util.*;
class GFG
{
   public static void main(String args[])
   {
      Scanner sc=new Scanner(System.in);
     //taking input from the user
     System.out.println("Enter the number");
       int num=sc.nextInt();
       int temp=num;//storing the number in a temporary variable
       int f=1,sum=0;
       while(num!=0)//running while loop until number becomes zero
       {
          f=1;
        //extracting last digit of the number
        //and storing in r
          int r=num%10;
        //for loop to find the factorial of a digit
          for(int i=1;i<=r;i++)
          {
             f=f*i;
          }
          sum=sum+f;//adding the factotial of the digits
          num=num/10;
       }
     //checking if the sum of the factorial of digits
     //is equal to the number or not
       if(sum==temp)
       System.out.println("PETERSON NUMBER");
       else
       System.out.println("NOT PETERSON NUMBER");
   }
   }
```

53. Write a Java Program to check whether twin prime or not.

Steps to check twin prime numbers

In order to check whether the numbers are twin prime or not, we have to follow the following steps:

Get pair of numbers from the user to check whether it is twin prime or not.
Check whether both the numbers are prime or not.
Find the difference between both numbers.
If both the numbers are prime and the difference of both the number is 2, print "numbers are the twin prime numbers".
If not, then print "numbers are not the twin prime numbers."
Let's implement the code to check whether the numbers are twin prime numbers or not.

```
//import required classes and packages
import java.io.*;
import java.util.*;

//create TwinPrimeNumbers class to check whether the numbers are twin prime or not
class TwinPrimeNumbers {

   // create checkPrimeNumber() method to check whether the number is prime or not
   static boolean checkPrimeNumber(int number)
   {
      int i;
      int m = 0;
      int flag = 0;
      m = number/2;
      if(number == 0 || number == 1){
         return false;
      }else{
         for(i = 2; i <= m ;i++){
            if(number%i == 0){
               flag=1;
               return false;
            }
         }
         if(flag == 0)
         {
            return true;
         }
      }
      return false;
   }

   // create checkTwinPrimeNumber() to check whether the numbers are twin prime or not
   static boolean checkTwinPrimeNumber(int number1, int number2)
```

```
    {
        if(checkPrimeNumber(number1) && checkPrimeNumber(number2) &&
Math.abs(number1 - number2) == 2)
            return true;
        else
            return false;
    }

    /* Driver program to test above function */
    public static void main(String[] args)
    {
         int number1, number2;

        //create scanner class object to get input from user
        Scanner sc = new Scanner(System.in);

        //show custom message
        System.out.println("Enter first number");

        //store user entered value into variable n1
        number1 = sc.nextInt();

        //show custom message
        System.out.println("Enter second number");

        //store user entered value into variable n2
        number2 = sc.nextInt();

        if (checkTwinPrimeNumber(number1, number2))
            System.out.println("("+number1 +", "+number2+ ") is a pair of twin
primes");
        else
            System.out.println("("+number1 +", "+number2+ ") is not a pair of twin
primes");

    }
}
```

54. Write a Java Program to check whether a number is Sunny Number or not.

Given, N=80 then N+1 will be 80+1=81, which is a perfect square of the number 9. Hence 80 is a sunny number.

Let's take another number 10.

Given, N=10 then N+1 will be 10+1=11, which is not a perfect square. Hence 10 is not a sunny number.

Steps to Find Sunny Number
The logic is very simple. To find the sunny number, we need only to check whether N+1 is the perfect square or not.

1. Read or initialize a number (num).
2. Add 1 to the given number i.e. num+1.
3. Find the square root of num+1.
4. If the square root is an integer, the given number is sunny, else not a sunny number.

```
import java.util.*;
public class SunnyNumberExample1
{
//driver code
public static void main(String args[])
{
Scanner sc = new Scanner(System.in);
System.out.print("Enter a number to check: ");
//reading a number from the user
int N=sc.nextInt();
//calling user-defined function
isSunnyNumber(N);
}
//function checks whether the given is a perfect square or not
static boolean findPerfectSquare(double num)
{
//finds the square root of the ggiven number
double square_root = Math.sqrt(num);
//if square root is an integer
return((square_root - Math.floor(square_root)) == 0);
}
//function that checks whether the given number is Sunny or not
static void isSunnyNumber(int N)
{
//checks N+1 is perfect square or not
if (findPerfectSquare(N + 1))
{
System.out.println("The given number is a sunny number.");
}
//executes if N+1 is not a perfect square
else
{
System.out.println("The given number is not a sunny number.");
```

```
}
}
}
```

55. Write a java program to check whether a number is divisible by 11 or not.

A number is divisible by 11 if the difference between the sum of its alternative digits is divisible by 11.

i.e. if (sum of odd digits) – (sum of even digits) is 0 or divisible by 11 then the given number is divisible by 11.

```
import java.util.Scanner;

public class DivisibleBy11 {
   public static void main(String args[]) {
      Scanner sc = new Scanner(System.in);
      System.out.println("Enter a number :");
      String num = sc.nextLine();
      int digitSumEve = 0;
      int digitSumOdd = 0;

     for(int i = 0; i<num.length(); i++) {
        if(i%2 == 0) {
           digitSumEve = digitSumEve + num.charAt(i)-'0';
        } else {
           digitSumOdd = digitSumOdd + num.charAt(i)-'0';
        }
      }
      int res = digitSumOdd-digitSumEve;
      if(res % 11 == 0) {
        System.out.println("Given number is divisible by 11");
      } else {
        System.out.println("Given number is not divisible by 11");
      }
   }
}
```

56. Write a java program to generate non fibonacci series.

```
import java.io.*;

class GFG {
    int[] holes = {21, 3, 6};
    int i = 0, j = 1, k, m, no;
```

```
    int[] b = new int[10];

    // Range is 10
    no = 10;
    b[1] = 0;
    b[2] = 1;

    // Check if range is less equals to 1
    if (no <= 1) {
        System.out.print("You have enter a wrong range");
    }

    // check if range is greater than 1
    // and less equals to 5
    else if (no <= 5 && no > 1) {
        System.out.print("\n" + "There is not any Non-Fibonacci series that lies between
1 to" + no +
        " term of Fibonacci Series.");
    }

    // If range is greater than 5
    else {

        // Loop to calculate fibonacci series till
        // range
        for (m = 2; m < no; m++) {
            k = i + j;
            i = j;
            j = k;

            // Store fibonacci series into b[]
            // array
            b[m] = k;
        }
        i = 5;
        System.out.println("\n" + "The Non-Fibonacci series that lies between 1 to "
             + no + " term of Fibonacci Series is: "+ "\n");

        // Loop to calculate Non-Fibonacci
        // series
        for (int ans = 4; ans < b[no - 1]; ans++) {
            if (ans != b[i])

                // Print Non-Fibonacci Series
                System.out.print(ans + "  ");
            else
```

```
            i++;
        }
    }
}
```

57. Write a java program to create and display singly linked list.

```
public class SinglyLinkedList {
    //Represent a node of the singly linked list
    class Node{
        int data;
        Node next;

        public Node(int data) {
            this.data = data;
            this.next = null;
        }
    }

    //Represent the head and tail of the singly linked list
    public Node head = null;
    public Node tail = null;

    //addNode() will add a new node to the list
    public void addNode(int data) {
        //Create a new node
        Node newNode = new Node(data);

        //Checks if the list is empty
        if(head == null) {
            //If list is empty, both head and tail will point to new node
            head = newNode;
            tail = newNode;
        }
        else {
            //newNode will be added after tail such that tail's next will point to newNode
            tail.next = newNode;
            //newNode will become new tail of the list
            tail = newNode;
        }
    }

    //display() will display all the nodes present in the list
    public void display() {
        //Node current will point to head
        Node current = head;
```

```
        if(head == null) {
            System.out.println("List is empty");
            return;
        }
        System.out.println("Nodes of singly linked list: ");
        while(current != null) {
            //Prints each node by incrementing pointer
            System.out.print(current.data + " ");
            current = current.next;
        }
        System.out.println();
    }

    public static void main(String[] args) {

        SinglyLinkedList sList = new SinglyLinkedList();

        //Add nodes to the list
        sList.addNode(1);
        sList.addNode(2);
        sList.addNode(3);
        sList.addNode(4);

        //Displays the nodes present in the list
        sList.display();
    }
}
```

58. Implement Queue Using Array in Java.

```
public class DemoQueue
{
  /* Member variable declaration */
  private int maxSize;
  private int[] queueArray;
  private int front;
  private int rear;
  private int currentSize;
  /* Constructor */
  public DemoQueue(int size)
  {
    this.maxSize = size;
    this.queueArray = new int[size];
    front = 0;
    rear = -1;
```

```
    currentSize = 0;
  }
  /* Queue:Insert Operation */
  public void insert(int item)
  {
    /* Checks whether the queue is full or not */
    if(isQueueFull())
    {
      System.out.println("Queue is full!");
      return;
    }
    if(rear == maxSize - 1)
{
      rear = -1;
    }
    /* increment rear then insert element to queue */
    queueArray[++rear] = item;
    currentSize++;
    System.out.println("Item added to queue: " + item);
  }
  /* Queue:Delete Operation */
  public int delete()
  {
    /* Checks whether the queue is empty or not */
    if(isQueueEmpty())
    {
      throw new RuntimeException("Queue is empty");
    }
    /* retrieve queue element then increment */
    int temp = queueArray[front++];
    if(front == maxSize)
    {
      front = 0;
    }
    currentSize--;
    return temp;
  }
  /* Queue:Peek Operation */
  public int peek()
  {
    return queueArray[front];
  }
  /* Queue:isFull Operation */
  public boolean isQueueFull()
  {
    return (maxSize == currentSize);
```

```
  }
  /* Queue:isEmpty Operation */
  public boolean isQueueEmpty()
{
    return (currentSize == 0);
  }
  /* Driver Code */
  public static void main(String[] args)
  {
    DemoQueue queue = new DemoQueue(10);
    queue.insert(2);
    queue.insert(3);
    System.out.println("Item deleted from queue: " + queue.delete());
    System.out.println("Item deleted from queue: " + queue.delete());
    queue.insert(5);
    System.out.println("Item deleted from queue: " + queue.delete());
  }
}
```

59. Stack Implementation using Linked List in Java.

```
/*
 * Java Program to Implement Stack using Linked List
 */

import java.util.*;

/*  Class Node  */
class Node
{
    protected int data;
    protected Node link;

    /*  Constructor  */
    public Node()
    {
        link = null;
        data = 0;
    }
    /*  Constructor  */
    public Node(int d,Node n)
    {
        data = d;
        link = n;
    }
    /*  Function to set link to next Node  */
```

```
    public void setLink(Node n)
    {
        link = n;
    }
    /*  Function to set data to current Node  */
    public void setData(int d)
    {
        data = d;
    }
    /*  Function to get link to next node  */
    public Node getLink()
    {
        return link;
    }
    /*  Function to get data from current Node  */
    public int getData()
    {
        return data;
    }
}

/*  Class linkedStack  */
class linkedStack
{
    protected Node top ;
    protected int size ;

    /*  Constructor  */
    public linkedStack()
    {
        top = null;
        size = 0;
    }
    /*  Function to check if stack is empty */
    public boolean isEmpty()
    {
        return top == null;
    }
    /*  Function to get the size of the stack */
    public int getSize()
    {
        return size;
    }
    /*  Function to push an element to the stack */
    public void push(int data)
    {
```

```
        Node nptr = new Node (data, null);
        if (top == null)
            top = nptr;
        else
        {
            nptr.setLink(top);
            top = nptr;
        }
        size++ ;
    }
    /*  Function to pop an element from the stack */
    public int pop()
    {
        if (isEmpty() )
            throw new NoSuchElementException("Underflow Exception") ;
        Node ptr = top;
        top = ptr.getLink();
        size-- ;
        return ptr.getData();
    }
    /*  Function to check the top element of the stack */
    public int peek()
    {
        if (isEmpty() )
            throw new NoSuchElementException("Underflow Exception") ;
        return top.getData();
    }
    /*  Function to display the status of the stack */
    public void display()
    {
        System.out.print("\nStack = ");
        if (size == 0)
        {
            System.out.print("Empty\n");
            return ;
        }
        Node ptr = top;
        while (ptr != null)
        {
            System.out.print(ptr.getData()+" ");
            ptr = ptr.getLink();
        }
        System.out.println();
    }
}
```

```
/* Class LinkedStackImplement */
public class LinkedStackImplement
{
    public static void main(String[] args)
    {
        Scanner scan = new Scanner(System.in);
        /* Creating object of class linkedStack */
        linkedStack ls = new linkedStack();
        /* Perform Stack Operations */
        System.out.println("Linked Stack Test\n");
        char ch;
        do
        {
            System.out.println("\nLinked Stack Operations");
            System.out.println("1. push");
            System.out.println("2. pop");
            System.out.println("3. peek");
            System.out.println("4. check empty");
            System.out.println("5. size");
            int choice = scan.nextInt();
            switch (choice)
            {
            case 1 :
                System.out.println("Enter integer element to push");
                ls.push( scan.nextInt() );
                break;
            case 2 :
                try
                {
                    System.out.println("Popped Element = "+ ls.pop());
                }
                catch (Exception e)
                {
                    System.out.println("Error : " + e.getMessage());
                }
                break;
            case 3 :
                try
                {
                    System.out.println("Peek Element = "+ ls.peek());
                }
                catch (Exception e)
                {
                    System.out.println("Error : " + e.getMessage());
                }
                break;
```

```
            case 4 :
               System.out.println("Empty status = "+ ls.isEmpty());
               break;
            case 5 :
               System.out.println("Size = "+ ls.getSize());
               break;
            case 6 :
               System.out.println("Stack = ");
               ls.display();
               break;
            default :
               System.out.println("Wrong Entry \n ");
               break;
            }
            /* display stack */
            ls.display();
            System.out.println("\nDo you want to continue (Type y or n) \n");
            ch = scan.next().charAt(0);

         } while (ch == 'Y'|| ch == 'y');
      }
}
```

60. Array implementation of Stack using Java.

```
import java.util.Scanner;
class Stack
{
   int top;
   int maxsize = 10;
   int[] arr = new int[maxsize];


   boolean isEmpty()
   {
      return (top < 0);
   }
   Stack()
   {
      top = -1;
   }
   boolean push (Scanner sc)
   {
      if(top == maxsize-1)
      {
         System.out.println("Overflow !!");
```

```
            return false;
        }
        else
        {
            System.out.println("Enter Value");
            int val = sc.nextInt();
            top++;
            arr[top]=val;
            System.out.println("Item pushed");
            return true;
        }
    }
    boolean pop ()
    {
        if (top == -1)
        {
            System.out.println("Underflow !!");
            return false;
        }
        else
        {
            top --;
            System.out.println("Item popped");
            return true;
        }
    }
    void display ()
    {
        System.out.println("Printing stack elements .....");
        for(int i = top; i>=0;i--)
        {
            System.out.println(arr[i]);
        }
    }
}
public class Stack_Operations {
public static void main(String[] args) {
    int choice=0;
    Scanner sc = new Scanner(System.in);
    Stack s = new Stack();
    System.out.println("*********Stack operations using array*********\n");
    System.out.println("\n-----------------------------------------------\n");
    while(choice != 4)
    {
        System.out.println("\nChose one from the below options...\n");
        System.out.println("\n1.Push\n2.Pop\n3.Show\n4.Exit");
```

```
        System.out.println("\n Enter your choice \n");
        choice = sc.nextInt();
        switch(choice)
        {
           case 1:
           {
              s.push(sc);
              break;
           }
           case 2:
           {
              s.pop();
              break;
           }
           case 3:
           {
              s.display();
              break;
           }
           case 4:
           {
              System.out.println("Exiting....");
              System.exit(0);
              break;
           }
           default:
           {
              System.out.println("Please Enter valid choice ");
           }
        };
     }
}
}
```

61. Write a java program to implement doubly linked list.

```
public class DoublyLinkedList {

   //Represent a node of the doubly linked list

   class Node{
      int data;
      Node previous;
      Node next;

      public Node(int data) {
```

```
        this.data = data;
    }
}

//Represent the head and tail of the doubly linked list
Node head, tail = null;

//addNode() will add a node to the list
public void addNode(int data) {
    //Create a new node
    Node newNode = new Node(data);

    //If list is empty
    if(head == null) {
        //Both head and tail will point to newNode
        head = tail = newNode;
        //head's previous will point to null
        head.previous = null;
        //tail's next will point to null, as it is the last node of the list
        tail.next = null;
    }
    else {
        //newNode will be added after tail such that tail's next will point to newNode
        tail.next = newNode;
        //newNode's previous will point to tail
        newNode.previous = tail;
        //newNode will become new tail
        tail = newNode;
        //As it is last node, tail's next will point to null
        tail.next = null;
    }
}

//display() will print out the nodes of the list
public void display() {
    //Node current will point to head
    Node current = head;
    if(head == null) {
        System.out.println("List is empty");
        return;
    }
    System.out.println("Nodes of doubly linked list: ");
    while(current != null) {
        //Prints each node by incrementing the pointer.

        System.out.print(current.data + " ");
```

```
            current = current.next;
        }
    }

    public static void main(String[] args) {

        DoublyLinkedList dList = new DoublyLinkedList();
        //Add nodes to the list
        dList.addNode(1);
        dList.addNode(2);
        dList.addNode(3);
        dList.addNode(4);
        dList.addNode(5);

        //Displays the nodes present in the list
        dList.display();
    }
}
```

62. Write a java program to add two matrices.

```
import java.util.Scanner;

class AddMatrix
{
public static void main(String args[])
{
int row, col,i,j;
Scanner in = new Scanner(System.in);

System.out.println("Enter the number of rows");
row = in.nextInt();

System.out.println("Enter the number columns");
col = in.nextInt();

int mat1[][] = new int[row][col];
int mat2[][] = new int[row][col];
int res[][] = new int[row][col];

System.out.println("Enter the elements of matrix1");

for ( i= 0 ; i < row ; i++ )
{

for ( j= 0 ; j < col ;j++ )
```

```
mat1[i][j] = in.nextInt();

System.out.println();
}
System.out.println("Enter the elements of matrix2");

for ( i= 0 ; i < row ; i++ )
{

for ( j= 0 ; j < col ;j++ )
mat2[i][j] = in.nextInt();

System.out.println();
}

for ( i= 0 ; i < row ; i++ )
for ( j= 0 ; j < col ;j++ )
res[i][j] = mat1[i][j] + mat2[i][j] ;

System.out.println("Sum of matrices:-");

for ( i= 0 ; i < row ; i++ )
{
for ( j= 0 ; j < col ;j++ )
System.out.print(res[i][j]+"\t");

System.out.println();
}

}
}
```

63. Java program to generate fibonacci series using recursion.

```
//Using Recursion
public class FibonacciCalc{
        public static int fibonacciRecursion(int n){
        if(n == 0){
                return 0;
        }
        if(n == 1 || n == 2){
                        return 1;
                }
        return fibonacciRecursion(n-2) + fibonacciRecursion(n-1);
        }
    public static void main(String args[]) {
```

```
        int maxNumber = 10;
        System.out.print("Fibonacci Series of "+maxNumber+" numbers: ");
        for(int i = 0; i < maxNumber; i++){
                        System.out.print(fibonacciRecursion(i) +" ");
                }
        }
}
```

64. Java program to calculate factorial using recursion.

```
class FactorialExample2{
 static int factorial(int n){
  if (n == 0)
    return 1;
  else
    return(n * factorial(n-1));
 }
 public static void main(String args[]){
  int i,fact=1;
  int number=4;//It is the number to calculate factorial
  fact = factorial(number);
  System.out.println("Factorial of "+number+" is: "+fact);
 }
}
```

65. Java Program to Find LCM and GCD of Two Numbers.

The Least Common Multiple (LCM) of two integers a and b, usually denoted by LCM (a, b), is the smallest positive integer that is divisible by both a and b.

The Highest Common Factor (HCF) of two or more integers, is the largest positive integer that divides the numbers without a remainder. HCF is also known as greatest common divisor(GCD) or greatest common factor(GCF).

If we know LCM or HCF of two numbers, then we can find the other one using below equation.

LCM(A, B) X HCF(A, B) = A*B

```
import java.util.Scanner;

/**
 * Java Program to print LCM and GCD of two numbers
 */
public class PrintLcmHcf {
```

```
    public static void main(String[] args) {
        int a, b, t, aTemp, bTemp, lcm, gcd;
        Scanner scanner;
        scanner = new Scanner(System.in);
        // Take two numbers from user
        System.out.println("Enter Two Number");
        a = scanner.nextInt();
        b = scanner.nextInt();

        aTemp = a;
        bTemp = b;

        while (bTemp != 0) {
            t = bTemp;
            bTemp = aTemp % bTemp;
            aTemp = t;
        }

        gcd = aTemp;

        /*
         * GCD(a, b) * LCM(a, b) = a*b
         */
        lcm = (a * b) / gcd;
        System.out.println("LCM = " + lcm);
        System.out.println("GCD = " + gcd);
    }
}
```

66. Java Program to Implement the sine() Function.

```
// Java Program to Implement the cos() Function(approximately)

import java.io.BufferedReader;
import java.io.InputStreamReader;

public class Sine {
    // Function to calculate and display sine of an angle
    public static void main(String[] args) {
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        double x;
        try {
            System.out.println("Enter the angle whose sine is to be
                                        calculated in degrees");
            x = Double.parseDouble(br.readLine());
        } catch (Exception e) {
```

```
            System.out.println("An error occurred");
            return;
        }
        double y;
        y = x*Math.PI/180;
        int n = 10;
        int i,j,fac;
        double sine = 0;
        for(i=0; i<=n; i++){
            fac = 1;
            for(j=2; j<=2*i+1; j++){
                fac*=j;
            }
            sine+=Math.pow(-1.0,i)*Math.pow(y,2*i+1)/fac;
        }
        System.out.format("The sine of " + x + " is %f",sine);
    }
}
```

67. Java Program to Implement the cosine() Function.

```
import java.lang.Math.*;

class sample
{
    static final double PI = 3.142;

    static double cosXSeriesSum(double x,
                                int n)
    {
         // here x is in degree.
        // we have to convert it to radian
        // for using it with series formula,
        // as in series expansion angle is in radian

        x = x * (PI / 180.0);

        double res = 1;
        double sign = 1, fact = 1,
                       pow = 1;
        for (int i = 1; i < 5; i++)
        {
            sign = sign * -1;
            fact = fact * (2 * i - 1) *
                          (2 * i);
            pow = pow * x * x;
```

```
            res = res + sign * pow / fact;
        }

        return res;
    }

    // Driver Code
    public static void main(String[] args)
    {
        float x = 50;
        int n = 5;
        System.out.println((float)(
            cosXSeriesSum(x, 5) * 1000000) /
                            1000000.00);
    }
}
```